# FLUJO DE ANÁLISIS DE DATOS 1: AJUSTANDO

*Juan Carlos Castillo / jc-castillo.com*

26 mayo, 2019

A continuación se presenta un ejemplo de flujo de análisis utilizando R / RStudio. Este flujo corresponde a la primera parte de ajuste de los datos, y más adelante se presentaran flujos de descripción y de modelamiento.

Actualmente existe mucha literatura y tutoriales sobre esto, pero en general se abordan aspectos del análisis por separado y con una profundidad y complejidad que en ocasiones es una barrera para utilizar los recursos de R de una manera más pragmática para análisis y reporte de datos.

El flujo presentado acá proviene de mi experiencia personal analizando datos y enseñando a hacerlo, y se orienta por la reproducibilidad y la parsimonia En relación a la reproducibilidad, se trata de siempre tener en mente que el código y reporte tienen que ser entendidos por otros y por mí mismo en el futuro. Y la parsimonia es parte de esto: la menor cantidad de recursos posibles para hacer lo más posible, y buscar las funciones más intuitivas.

## *Librerías y datos*

### *Librerías*

El procedimiento de análisis en R requiere usualmente cargar librerías, que son un conjunto de funciones para análisis específicos. Las librerías para análisis básicos vienen pre-instaladas, y para funciones más específicas hay que instalarla (en caso de no estar ya instalada), y si ya está instalada se carga. Ejemplo, la librería `dplyr`

```
install.packages("dplyr")  # para instalar
library(dplyr)  # para cargar
```

Para hacer más expedito este procedimiento se recomienda utilizar la librería `pacman`, que sirve para instalar y cargar de manera expedita otras librerías. También se recomienda comentar luego de cada librería para qué se va a utilizar, para llevar un registro [1]

[1] `pacman::p_load(library1,library2...)`

```
pacman::p_load(haven, # abrir bases de datos en otros formatos
dplyr, # data management
car, # recode
sjmisc # descriptivos
)
```

Recomendación general: cargar en el preámbulo las librerías que más se utilizan durante el análisis. Si alguna función de una librería

se utiliza pocas veces, es mejor utilizar una segunda forma de acceder a las funciones sin necesidad de cargar la librería. Por ejemplo, si quiero utilizar la función para recodificar `rec` de la librería `sjmisc`:

```
sjmisc::rec(x, 1=2)
```

En este caso, la lógica es librería::función, y no se requiere cargar la librería pero sí *debe estar previamente instalada*.

### *Abrir base de datos*

En caso que las bases estén en formatos separados por comas u otro separador (tabulaciones, espacios):

```
data <- read.table("ruta/datos.csv", sep = ",", header = TRUE)
```

Si se encuentran en un formato estándar tipo spss, stata, sas, se recomienda utilizar las funciones respectivas de la librería `haven`. En este caso, la base de datos a utilizar (ELSOC) se encuentra en formato Stata, y utilizamos la función `read_dta`. Se da la ruta al archivo local, en este caso la llamamos directamente de la web

```
data <- read_dta("https://juancarloscastillo.github.io/metsoc-facsouchile/documents/data/COESW1_Stata14_V3
```

Y realizamos un chequeo básico de la lectura de datos: nombres de las variables y tamaño de la base en términos de casos y variables (en este ejemplo, 2984 casos y 374 variables).

```
dim(data)  # dimension de la base
```

```
## [1] 2984  374
```

Y si se quiere revisar en formato de planilla de datos:

```
View(data)
```

# *Ajuste y generación de variables*

Esta parte inicial del análisis es usualmente la más tediosa y la más larga, y consiste en dejar los datos listos para ser analizados. Los procedimientos usuales son selección y renombramiento de las variables, identificación de casos perdidos, recodificaciones y generación de índices simples.

## *Selección y renombre de variables*

Este paso es opcional y consiste en crear un subset de datos para continuar con los análisis, en lugar de hacerlo con la base completa. Para ello:

- Primero, se identifica el nombre de las variables. Esto aparece en el libro de códigos y/o en el cuestionario, o también se puede hacer buscando en la base de datos mediante alguna palabra clave asociada a la pregunta. Por ejemplo, si queremos buscar variables asociadas a educación, utilizamos la función `find_var`, de `sjmisc` (entrega nombre de la variable en columna var.name)

```
find_var(data, "edu")
```

```
##   col.nr var.name                                                                  var.label
## 1    197    d2_02 Es justo que las personas de altos ingresos tengan una mejor educacion para sus
## 2    204    d5_02                                                 Tener un buen nivel de educacion
## 3    242    f5_07 Que estudiantes tiren piedras a Carabineros en una marcha por la educacion del p
## 4    275       m1 Cual es su nivel educacional? Indique el tipo de estudio actual (si estudia actu
## 5    295      m20                Cual es el nivel educacional del principal sostenedor del hogar?
## 6    302      m27                                    Cual es/fue el nivel educacional de su padre?
## 7    303      m28                                    Cual es/fue el nivel educacional de su madre?
```

- En segundo lugar se crea un subset que contiene solo las variables que se van a utilizar en el análisis, mediante la función `select` de `dplyr`. Con esta función se puede además en un paso otorgar un nombre más sustantivo a la variable, que facilite su posterior utilización. En este caso, daremos el nombre sstatus a la variable

de estatus subjetivo d1_01, ingreso a m30, educación a m1, sexo se queda con su nombre y nhogar (número de personas del hogar) también. Este subset puede cambiar posteriormente, solo agregar/quitar variables y correr el código nuevamente

```
data_n <- data %>% select(sstatus = d1_01, ingreso_monto = m29monto,
    ingreso_rango = m30, educacion = m1, sexo, nhogar)
summary(data_n)
```

```
##     sstatus      ingreso_monto       ingreso_rango     educacion           sexo           nhogar
##  Min.   : 0.00   Min.   :0.000e+00   Min.   : 1.00   Min.   : 1.000   Min.   :1.000   Min.   : 1.00
##  1st Qu.: 3.00   1st Qu.:1.000e+05   1st Qu.: 4.00   1st Qu.: 4.000   1st Qu.:1.000   1st Qu.: 2.00
##  Median : 5.00   Median :3.300e+05   Median : 9.00   Median : 5.000   Median :2.000   Median : 3.00
##  Mean   : 4.68   Mean   :1.945e+06   Mean   :28.65   Mean   : 5.327   Mean   :1.606   Mean   : 3.09
##  3rd Qu.: 5.00   3rd Qu.:6.000e+05   3rd Qu.:20.00   3rd Qu.: 7.000   3rd Qu.:2.000   3rd Qu.: 4.00
##  Max.   :99.00   Max.   :4.000e+09   Max.   :99.00   Max.   :99.000   Max.   :2.000   Max.   :15.00
##                                      NA's   :2350
```

## *Recodificación de variables*

**Introducción**

En el caso de trabajar con R, el ajuste de las variables pasa por atender el distinto tipo de estructura, usualmente numérico (vector) o variable categórica (factor). Esta definición establece diferencias claras; por ejemplo, no se puede hacer un promedio con un factor, y los vectores númericos no tienen etiquetas. Dado que el límite entre lo categórico y lo continuo muchas veces es más relativo en ciencias sociales, para algunas operaciones resulta más eficiente contar con variables numéricas que puedan tener etiquetas, y que luego puedan ser transformadas a factor para operaciones específicas. En R este formato intermedio se denomina "vector atómico" o "vector etiquetado".

En primer lugar revisaremos la estructura de las variables de la base de datos con el comando `str` (estructura)

```
str(data_n)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    2984 obs. of  6 variables:
##  $ sstatus      : 'haven_labelled' num  5 5 3 6 4 5 4 6 7 4 ...
##   ..- attr(*, "label")= chr "Donde se ubicaria usted en la sociedad chilena?"
##   ..- attr(*, "format.stata")= chr "%17.0g"
##   ..- attr(*, "labels")= Named num  0 10 88 99
##   .. ..- attr(*, "names")= chr  "El nivel mas bajo" "El nivel mas alto" "No sabe" "No responde"
##  $ ingreso_monto: num  3e+05 5e+05 9e+00 8e+00 9e+00 8e+00 9e+00 5e+05 9e+00 3e+05 ...
##   ..- attr(*, "label")= chr "En el mes pasado, cual fue el ingreso total de su hogar? (Considere los in
```

```
##   ..- attr(*, "format.stata")= chr "%10.0g"
##  $ ingreso_rango: 'haven_labelled' num  NA NA 7 12 7 7 12 NA 13 NA ...
##   ..- attr(*, "label")= chr "A continuacion le presentamos un listado de rangos de ingreso. Podria uste
##   ..- attr(*, "format.stata")= chr "%45.0g"
##   ..- attr(*, "labels")= Named num  1 2 3 4 5 6 7 8 9 10 ...
##   .. ..- attr(*, "names")= chr  "Menos de $220.000 mensuales liquidos" "De $220.001 a $280.000 mensuale
##  $ educacion    : 'haven_labelled' num  2 4 4 8 5 4 3 5 6 5 ...
##   ..- attr(*, "label")= chr "Cual es su nivel educacional? Indique el tipo de estudio actual (si estudi
##   ..- attr(*, "format.stata")= chr "%43.0g"
##   ..- attr(*, "labels")= Named num  1 2 3 4 5 6 7 8 9 10 ...
##   .. ..- attr(*, "names")= chr  "Sin estudios" "Educacion Basica o Preparatoria incompleta" "Educacion
##  $ sexo         : 'haven_labelled' num  2 2 2 1 1 1 1 2 2 1 ...
##   ..- attr(*, "label")= chr "Sexo del entrevistado"
##   ..- attr(*, "format.stata")= chr "%9.0g"
##   ..- attr(*, "labels")= Named num  1 2
##   .. ..- attr(*, "names")= chr  "Hombre" "Mujer"
##  $ nhogar       : num  4 2 6 1 1 1 1 2 3 4 ...
##   ..- attr(*, "label")= chr "Numero de miembros del hogar"
##   ..- attr(*, "format.stata")= chr "%9.0g"
```

En este caso, vemos que las cuatro vectores se definen como "haven_labelled", que hace referencia a vectores numéricos etiquetados. Esto ocurre en el proceso de conversión mediante el paquete `haven`, que transforma de esta manera a los valores numéricos con etiqueta que vienen de Stata.

Los vectores que que aparecen aquí se caracterizan por una serie de atributos (attr):

- label: etiqueta de la variable
- format.stata: formato de la etiqueta de la variable
- labels: niveles de respuesta de la variable (values) que están numerados
- names: nombre de las etiquetas de los valores.

Estos atributos luego van a facilitar algunas operaciones de recodificación y de descripción de los datos.

**Proceso de recodificación**

Para que esto sea lo más claro y reproducible posible, se recomienda hacer este procedimiento variable por variable, con la siguiente secuencia: - descripción inicial - transformaciones necesarias - descripción y chequeo final.

Si bien hay una serie de librerías de R asociadas a esto, optamos por la librería `sjmisc` (cargada en el inicio), en primer lugar para minimizar el número de librerías y funciones asociadas, en segundo lugar por su versatilidad en el uso de etiquetas, y finalmente por su

compatibilidad con `dplyr`. La librería `sjmisc` también tiene una hoja resumen que se puede encontrar aquí.

- Variable dependiente: Estatus subjetivo

  Descripción inicial de la variable:

```
frq(data_n$sstatus)
```

```
##
## # Donde se ubicaria usted en la sociedad chilena? (x) <numeric>
## # total N=2984  valid N=2984  mean=4.68  sd=5.58
##
##  val             label frq raw.prc valid.prc cum.prc
##    0 El nivel mas bajo  46    1.54      1.54    1.54
##    1                 1  90    3.02      3.02    4.56
##    2                 2 210    7.04      7.04   11.60
##    3                 3 440   14.75     14.75   26.34
##    4                 4 682   22.86     22.86   49.20
##    5                 5 987   33.08     33.08   82.27
##    6                 6 329   11.03     11.03   93.30
##    7                 7 122    4.09      4.09   97.39
##    8                 8  40    1.34      1.34   98.73
##    9                 9   4    0.13      0.13   98.86
##   10 El nivel mas alto  22    0.74      0.74   99.60
##   88           No sabe  11    0.37      0.37   99.97
##   99       No responde   1    0.03      0.03  100.00
##   NA                NA   0    0.00        NA      NA
```

Vemos que los valores 88 y 99 corresponde recodificarlos a valores perdidos, que en el caso de R se define como `NA`. Lo hacemos mediante la funcion `recode` de la librería `car` (Nota: dplyr, sjmisc y otras librerías tienen su propia versión de recodificación, pero la de car es más intuitiva)

```
data_n$sstatus <- recode(data_n$sstatus, "c(88,99)=NA")
frq(data_n$sstatus)  # check
```

```
##
## # Donde se ubicaria usted en la sociedad chilena? (x) <numeric>
## # total N=2984  valid N=2972  mean=4.34  sd=1.58
##
##  val             label frq raw.prc valid.prc cum.prc
##    0 El nivel mas bajo  46    1.54      1.55    1.55
##    1                 1  90    3.02      3.03    4.58
##    2                 2 210    7.04      7.07   11.64
```

```
##    3                3 440   14.75     14.80   26.45
##    4                4 682   22.86     22.95   49.39
##    5                5 987   33.08     33.21   82.60
##    6                6 329   11.03     11.07   93.67
##    7                7 122    4.09      4.10   97.78
##    8                8  40    1.34      1.35   99.13
##    9                9   4    0.13      0.13   99.26
##   10 El nivel mas alto  22    0.74      0.74  100.00
##   88           No sabe   0    0.00      0.00  100.00
##   99       No responde   0    0.00      0.00  100.00
##   NA                NA  12    0.40        NA      NA
```

- Variable indepediente Ingreso

Esta variable en general tiene dos formas de preguntarse: en el monto directo en pesos, o en rangos de ingreso. Hay una tercera variante donde se pregunta por el monto directo, y quienes no responden luego se les pregunta por el rango. Este último es el caso de ELSOC. Como estas variables en general tienen muchos datos perdidos, para poder aprovechar la información lo ideal es combinarlas. Una posibilidad es la siguiente:

1 - crear una variable de ingresos en pesos a partir de la pregunta de rangos, mediante la imputación del promedio del rango 2 - combinar las variables de ingreso (ing_comb) 3 - para tener mayor sensibilidad a las implicancias del ingreso, crear una variable de ingreso equivalente por hogar, dividiendo la variable de ingreso por el número de personas del hogar.

*ingreso_monto:*

```
frq(data_n$ingreso_monto)
```

```
##
## # En el mes pasado, cual fue el ingreso total de su hogar? (Considere los ingresos (x) <numeric>
## # total N=2984  valid N=2984  mean=1945485.61  sd=73385240.89
##
##       val frq raw.prc valid.prc cum.prc
##         0   8    0.27      0.27    0.27
##         8 410   13.74     13.74   14.01
##         9 224    7.51      7.51   21.51
##     20000   1    0.03      0.03   21.55
##     21000   1    0.03      0.03   21.58
##     24000   1    0.03      0.03   21.62
##     25000   1    0.03      0.03   21.65
##     30000   3    0.10      0.10   21.75
##     40000   2    0.07      0.07   21.82
##     50000   6    0.20      0.20   22.02
```

```
##     55000   2    0.07      0.07   22.08
##     60000   7    0.23      0.23   22.32
##     70000   2    0.07      0.07   22.39
##     76000   1    0.03      0.03   22.42
##     80000   4    0.13      0.13   22.55
##     81000   1    0.03      0.03   22.59
##     83000   1    0.03      0.03   22.62
##     86000   2    0.07      0.07   22.69
##     87000   1    0.03      0.03   22.72
##     88000   1    0.03      0.03   22.75
##     89000  13    0.44      0.44   23.19
##     89265   1    0.03      0.03   23.22
##     89400   1    0.03      0.03   23.26
##     89740   1    0.03      0.03   23.29
##     89900   1    0.03      0.03   23.32
##     90000  11    0.37      0.37   23.69
##     93000   2    0.07      0.07   23.76
##     96000   1    0.03      0.03   23.79
##     97000   2    0.07      0.07   23.86
##     98000   2    0.07      0.07   23.93
##      1e+05  34    1.14      1.14   25.07
##    110000   3    0.10      0.10   25.17
##    115000   1    0.03      0.03   25.20
##    116940   1    0.03      0.03   25.23
##    117000   2    0.07      0.07   25.30
##    118000   1    0.03      0.03   25.34
##    119000   1    0.03      0.03   25.37
##    120000  29    0.97      0.97   26.34
##    122000   1    0.03      0.03   26.37
##    125000   3    0.10      0.10   26.47
##    130000  11    0.37      0.37   26.84
##    136000   1    0.03      0.03   26.88
##    140000   5    0.17      0.17   27.04
##    141000   1    0.03      0.03   27.08
##    147000   1    0.03      0.03   27.11
##    148000   1    0.03      0.03   27.14
##    150000  37    1.24      1.24   28.38
##    154000   1    0.03      0.03   28.42
##    155000   1    0.03      0.03   28.45
##    160000  15    0.50      0.50   28.95
##    170000   5    0.17      0.17   29.12
##    173000   1    0.03      0.03   29.16
##    176000   1    0.03      0.03   29.19
##    178000   1    0.03      0.03   29.22
```

```
##    180000  23    0.77      0.77   29.99
##    186000   2    0.07      0.07   30.06
##    190000   9    0.30      0.30   30.36
##     2e+05 117    3.92      3.92   34.28
##    200008   1    0.03      0.03   34.32
##    208000   1    0.03      0.03   34.35
##    210000   5    0.17      0.17   34.52
##    212000   1    0.03      0.03   34.55
##    213000   1    0.03      0.03   34.58
##    215000   2    0.07      0.07   34.65
##    216000   1    0.03      0.03   34.68
##    220000  11    0.37      0.37   35.05
##    221000   1    0.03      0.03   35.09
##    225000   1    0.03      0.03   35.12
##    230000   8    0.27      0.27   35.39
##    235000   3    0.10      0.10   35.49
##    236000   1    0.03      0.03   35.52
##    240000  19    0.64      0.64   36.16
##    241000   1    0.03      0.03   36.19
##    244000   1    0.03      0.03   36.23
##    245000   2    0.07      0.07   36.29
##    250000 114    3.82      3.82   40.11
##    257000   4    0.13      0.13   40.25
##    257500   1    0.03      0.03   40.28
##    258000   3    0.10      0.10   40.38
##    260000  12    0.40      0.40   40.78
##    265000   2    0.07      0.07   40.85
##    266000   1    0.03      0.03   40.88
##    268000   1    0.03      0.03   40.92
##    270000  18    0.60      0.60   41.52
##    275000   2    0.07      0.07   41.59
##    280000  30    1.01      1.01   42.59
##    285000   1    0.03      0.03   42.63
##    289000   2    0.07      0.07   42.69
##    290000   4    0.13      0.13   42.83
##     3e+05 189    6.33      6.33   49.16
##    310000   5    0.17      0.17   49.33
##    318000   1    0.03      0.03   49.36
##    320000  17    0.57      0.57   49.93
##    330000  10    0.34      0.34   50.27
##    333000   1    0.03      0.03   50.30
##    340000   5    0.17      0.17   50.47
##    350000  77    2.58      2.58   53.05
##    354000   1    0.03      0.03   53.08
```

```
##    360000   8    0.27      0.27   53.35
##    370000   6    0.20      0.20   53.55
##    375000   1    0.03      0.03   53.59
##    380000  17    0.57      0.57   54.16
##    390000   2    0.07      0.07   54.22
##    397000   1    0.03      0.03   54.26
##     4e+05 187    6.27      6.27   60.52
##    420000   6    0.20      0.20   60.72
##    421000   1    0.03      0.03   60.76
##    430000   6    0.20      0.20   60.96
##    438000   1    0.03      0.03   60.99
##    440000   2    0.07      0.07   61.06
##    445000   1    0.03      0.03   61.09
##    450000  66    2.21      2.21   63.30
##    460000   2    0.07      0.07   63.37
##    465000   1    0.03      0.03   63.40
##    470000   6    0.20      0.20   63.61
##    480000  17    0.57      0.57   64.18
##    490000   2    0.07      0.07   64.24
##    496000   1    0.03      0.03   64.28
##     5e+05 205    6.87      6.87   71.15
##    510000   1    0.03      0.03   71.18
##    520000   3    0.10      0.10   71.28
##    522000   1    0.03      0.03   71.31
##    540000   5    0.17      0.17   71.48
##    550000  31    1.04      1.04   72.52
##    560000   2    0.07      0.07   72.59
##    570000   2    0.07      0.07   72.65
##    580000  11    0.37      0.37   73.02
##    585000   1    0.03      0.03   73.06
##    588000   1    0.03      0.03   73.09
##    590000   1    0.03      0.03   73.12
##     6e+05 134    4.49      4.49   77.61
##    613000   1    0.03      0.03   77.65
##    620000   2    0.07      0.07   77.71
##    630000   2    0.07      0.07   77.78
##    650000  27    0.90      0.90   78.69
##    660000   1    0.03      0.03   78.72
##    665000   1    0.03      0.03   78.75
##    665001   1    0.03      0.03   78.79
##    680000   3    0.10      0.10   78.89
##    690000   1    0.03      0.03   78.92
##     7e+05 101    3.38      3.38   82.31
##    720000   1    0.03      0.03   82.34
```

```
##    730000   1    0.03      0.03   82.37
##    737000   1    0.03      0.03   82.41
##    740000   2    0.07      0.07   82.47
##    750000  17    0.57      0.57   83.04
##    760000   2    0.07      0.07   83.11
##    770000   1    0.03      0.03   83.14
##    775000   1    0.03      0.03   83.18
##    780000   2    0.07      0.07   83.24
##     8e+05 110    3.69      3.69   86.93
##    850000  10    0.34      0.34   87.27
##    860000   1    0.03      0.03   87.30
##    890000   1    0.03      0.03   87.33
##     9e+05  42    1.41      1.41   88.74
##    930000   1    0.03      0.03   88.77
##    950000   5    0.17      0.17   88.94
##    970000   1    0.03      0.03   88.97
##    980000   1    0.03      0.03   89.01
##     1e+06  82    2.75      2.75   91.76
##   1050000   1    0.03      0.03   91.79
##   1100000   8    0.27      0.27   92.06
##   1150000   1    0.03      0.03   92.09
##   1190000   1    0.03      0.03   92.12
##   1200000  35    1.17      1.17   93.30
##   1250000   1    0.03      0.03   93.33
##   1300000  15    0.50      0.50   93.83
##   1350000   1    0.03      0.03   93.87
##   1400000  12    0.40      0.40   94.27
##   1460000   1    0.03      0.03   94.30
##   1470000   1    0.03      0.03   94.34
##   1500000  35    1.17      1.17   95.51
##   1600000  16    0.54      0.54   96.05
##   1700000   1    0.03      0.03   96.08
##   1800000  15    0.50      0.50   96.58
##   1900000   1    0.03      0.03   96.62
##     2e+06  30    1.01      1.01   97.62
##   2200000   3    0.10      0.10   97.72
##   2300000   1    0.03      0.03   97.75
##   2400000   3    0.10      0.10   97.86
##   2500000  16    0.54      0.54   98.39
##   2800000   1    0.03      0.03   98.42
##     3e+06  17    0.57      0.57   98.99
##   3500000   4    0.13      0.13   99.13
##     4e+06   7    0.23      0.23   99.36
##   4400000   1    0.03      0.03   99.40
```

```
##     5e+06   4    0.13      0.13   99.53
##   5500000   2    0.07      0.07   99.60
##     6e+06   3    0.10      0.10   99.70
##     8e+06   2    0.07      0.07   99.77
##   8500000   1    0.03      0.03   99.80
##     1e+07   2    0.07      0.07   99.87
##   2.3e+07   1    0.03      0.03   99.90
##  1.04e+08   1    0.03      0.03   99.93
##   2.5e+08   1    0.03      0.03   99.97
##     4e+09   1    0.03      0.03  100.00
##      <NA>   0    0.00        NA      NA
```

Recodificar 0, 8 y 9 como perdidos:

```
data_n$ingreso_monto <- recode(data_n$ingreso_monto, "c(0,8,9)=NA")
frq(data_n$ingreso_monto)  # check ok
```

```
##
## # En el mes pasado, cual fue el ingreso total de su hogar? (Considere los ingresos (x) <numeric>
## # total N=2984  valid N=2342  mean=2478788.96  sd=82830978.38
##
##       val frq raw.prc valid.prc cum.prc
##     20000   1    0.03      0.04    0.04
##     21000   1    0.03      0.04    0.09
##     24000   1    0.03      0.04    0.13
##     25000   1    0.03      0.04    0.17
##     30000   3    0.10      0.13    0.30
##     40000   2    0.07      0.09    0.38
##     50000   6    0.20      0.26    0.64
##     55000   2    0.07      0.09    0.73
##     60000   7    0.23      0.30    1.02
##     70000   2    0.07      0.09    1.11
##     76000   1    0.03      0.04    1.15
##     80000   4    0.13      0.17    1.32
##     81000   1    0.03      0.04    1.37
##     83000   1    0.03      0.04    1.41
##     86000   2    0.07      0.09    1.49
##     87000   1    0.03      0.04    1.54
##     88000   1    0.03      0.04    1.58
##     89000  13    0.44      0.56    2.13
##     89265   1    0.03      0.04    2.18
##     89400   1    0.03      0.04    2.22
##     89740   1    0.03      0.04    2.26
##     89900   1    0.03      0.04    2.31
##     90000  11    0.37      0.47    2.78
```

```
##     93000   2    0.07      0.09    2.86
##     96000   1    0.03      0.04    2.90
##     97000   2    0.07      0.09    2.99
##     98000   2    0.07      0.09    3.07
##     1e+05  34    1.14      1.45    4.53
##    110000   3    0.10      0.13    4.65
##    115000   1    0.03      0.04    4.70
##    116940   1    0.03      0.04    4.74
##    117000   2    0.07      0.09    4.82
##    118000   1    0.03      0.04    4.87
##    119000   1    0.03      0.04    4.91
##    120000  29    0.97      1.24    6.15
##    122000   1    0.03      0.04    6.19
##    125000   3    0.10      0.13    6.32
##    130000  11    0.37      0.47    6.79
##    136000   1    0.03      0.04    6.83
##    140000   5    0.17      0.21    7.05
##    141000   1    0.03      0.04    7.09
##    147000   1    0.03      0.04    7.13
##    148000   1    0.03      0.04    7.17
##    150000  37    1.24      1.58    8.75
##    154000   1    0.03      0.04    8.80
##    155000   1    0.03      0.04    8.84
##    160000  15    0.50      0.64    9.48
##    170000   5    0.17      0.21    9.69
##    173000   1    0.03      0.04    9.74
##    176000   1    0.03      0.04    9.78
##    178000   1    0.03      0.04    9.82
##    180000  23    0.77      0.98   10.80
##    186000   2    0.07      0.09   10.89
##    190000   9    0.30      0.38   11.27
##     2e+05 117    3.92      5.00   16.27
##    200008   1    0.03      0.04   16.31
##    208000   1    0.03      0.04   16.35
##    210000   5    0.17      0.21   16.57
##    212000   1    0.03      0.04   16.61
##    213000   1    0.03      0.04   16.65
##    215000   2    0.07      0.09   16.74
##    216000   1    0.03      0.04   16.78
##    220000  11    0.37      0.47   17.25
##    221000   1    0.03      0.04   17.29
##    225000   1    0.03      0.04   17.34
##    230000   8    0.27      0.34   17.68
##    235000   3    0.10      0.13   17.81
```

```
##    236000   1    0.03      0.04   17.85
##    240000  19    0.64      0.81   18.66
##    241000   1    0.03      0.04   18.70
##    244000   1    0.03      0.04   18.74
##    245000   2    0.07      0.09   18.83
##    250000 114    3.82      4.87   23.70
##    257000   4    0.13      0.17   23.87
##    257500   1    0.03      0.04   23.91
##    258000   3    0.10      0.13   24.04
##    260000  12    0.40      0.51   24.55
##    265000   2    0.07      0.09   24.64
##    266000   1    0.03      0.04   24.68
##    268000   1    0.03      0.04   24.72
##    270000  18    0.60      0.77   25.49
##    275000   2    0.07      0.09   25.58
##    280000  30    1.01      1.28   26.86
##    285000   1    0.03      0.04   26.90
##    289000   2    0.07      0.09   26.99
##    290000   4    0.13      0.17   27.16
##     3e+05 189    6.33      8.07   35.23
##    310000   5    0.17      0.21   35.44
##    318000   1    0.03      0.04   35.48
##    320000  17    0.57      0.73   36.21
##    330000  10    0.34      0.43   36.64
##    333000   1    0.03      0.04   36.68
##    340000   5    0.17      0.21   36.89
##    350000  77    2.58      3.29   40.18
##    354000   1    0.03      0.04   40.22
##    360000   8    0.27      0.34   40.56
##    370000   6    0.20      0.26   40.82
##    375000   1    0.03      0.04   40.86
##    380000  17    0.57      0.73   41.59
##    390000   2    0.07      0.09   41.67
##    397000   1    0.03      0.04   41.72
##     4e+05 187    6.27      7.98   49.70
##    420000   6    0.20      0.26   49.96
##    421000   1    0.03      0.04   50.00
##    430000   6    0.20      0.26   50.26
##    438000   1    0.03      0.04   50.30
##    440000   2    0.07      0.09   50.38
##    445000   1    0.03      0.04   50.43
##    450000  66    2.21      2.82   53.25
##    460000   2    0.07      0.09   53.33
##    465000   1    0.03      0.04   53.37
```

```
##    470000   6    0.20      0.26   53.63
##    480000  17    0.57      0.73   54.36
##    490000   2    0.07      0.09   54.44
##    496000   1    0.03      0.04   54.48
##     5e+05 205    6.87      8.75   63.24
##    510000   1    0.03      0.04   63.28
##    520000   3    0.10      0.13   63.41
##    522000   1    0.03      0.04   63.45
##    540000   5    0.17      0.21   63.66
##    550000  31    1.04      1.32   64.99
##    560000   2    0.07      0.09   65.07
##    570000   2    0.07      0.09   65.16
##    580000  11    0.37      0.47   65.63
##    585000   1    0.03      0.04   65.67
##    588000   1    0.03      0.04   65.71
##    590000   1    0.03      0.04   65.76
##     6e+05 134    4.49      5.72   71.48
##    613000   1    0.03      0.04   71.52
##    620000   2    0.07      0.09   71.61
##    630000   2    0.07      0.09   71.69
##    650000  27    0.90      1.15   72.84
##    660000   1    0.03      0.04   72.89
##    665000   1    0.03      0.04   72.93
##    665001   1    0.03      0.04   72.97
##    680000   3    0.10      0.13   73.10
##    690000   1    0.03      0.04   73.14
##     7e+05 101    3.38      4.31   77.46
##    720000   1    0.03      0.04   77.50
##    730000   1    0.03      0.04   77.54
##    737000   1    0.03      0.04   77.58
##    740000   2    0.07      0.09   77.67
##    750000  17    0.57      0.73   78.39
##    760000   2    0.07      0.09   78.48
##    770000   1    0.03      0.04   78.52
##    775000   1    0.03      0.04   78.57
##    780000   2    0.07      0.09   78.65
##     8e+05 110    3.69      4.70   83.35
##    850000  10    0.34      0.43   83.77
##    860000   1    0.03      0.04   83.82
##    890000   1    0.03      0.04   83.86
##     9e+05  42    1.41      1.79   85.65
##    930000   1    0.03      0.04   85.70
##    950000   5    0.17      0.21   85.91
##    970000   1    0.03      0.04   85.95
```

```
##    980000   1    0.03      0.04   85.99
##     1e+06  82    2.75      3.50   89.50
##   1050000   1    0.03      0.04   89.54
##   1100000   8    0.27      0.34   89.88
##   1150000   1    0.03      0.04   89.92
##   1190000   1    0.03      0.04   89.97
##   1200000  35    1.17      1.49   91.46
##   1250000   1    0.03      0.04   91.50
##   1300000  15    0.50      0.64   92.14
##   1350000   1    0.03      0.04   92.19
##   1400000  12    0.40      0.51   92.70
##   1460000   1    0.03      0.04   92.74
##   1470000   1    0.03      0.04   92.78
##   1500000  35    1.17      1.49   94.28
##   1600000  16    0.54      0.68   94.96
##   1700000   1    0.03      0.04   95.00
##   1800000  15    0.50      0.64   95.64
##   1900000   1    0.03      0.04   95.69
##     2e+06  30    1.01      1.28   96.97
##   2200000   3    0.10      0.13   97.10
##   2300000   1    0.03      0.04   97.14
##   2400000   3    0.10      0.13   97.27
##   2500000  16    0.54      0.68   97.95
##   2800000   1    0.03      0.04   97.99
##     3e+06  17    0.57      0.73   98.72
##   3500000   4    0.13      0.17   98.89
##     4e+06   7    0.23      0.30   99.19
##   4400000   1    0.03      0.04   99.23
##     5e+06   4    0.13      0.17   99.40
##   5500000   2    0.07      0.09   99.49
##     6e+06   3    0.10      0.13   99.62
##     8e+06   2    0.07      0.09   99.70
##   8500000   1    0.03      0.04   99.74
##     1e+07   2    0.07      0.09   99.83
##   2.3e+07   1    0.03      0.04   99.87
##  1.04e+08   1    0.03      0.04   99.91
##   2.5e+08   1    0.03      0.04   99.96
##     4e+09   1    0.03      0.04  100.00
##      <NA> 642   21.51        NA      NA
```

Esta variable queda con 642 NA, de los que se espera rescatar información mediante la variable de ingreso_rango

*ingreso_rango*

```
frq(data_n$ingreso_rango)
```

```
##
## # A continuacion le presentamos un listado de rangos de ingreso. Podria usted indi (x) <numeric>
## # total N=2984  valid N=634  mean=28.65  sd=38.07
##
##  val                                      label  frq raw.prc valid.prc cum.prc
##    1          Menos de $220.000 mensuales liquidos   51    1.71      8.04    8.04
##    2     De $220.001 a $280.000 mensuales liquidos   44    1.47      6.94   14.98
##    3     De $280.001 a $330.000 mensuales liquidos   39    1.31      6.15   21.14
##    4     De $330.001 a $380.000 mensuales liquidos   39    1.31      6.15   27.29
##    5     De $380.001 a $420.000 mensuales liquidos   37    1.24      5.84   33.12
##    6     De $420.001 a $470.000 mensuales liquidos   37    1.24      5.84   38.96
##    7     De $470.001 a $510.000 mensuales liquidos   41    1.37      6.47   45.43
##    8     De $510.001 a $560.000 mensuales liquidos   22    0.74      3.47   48.90
##    9     De $560.001 a $610.000 mensuales liquidos   26    0.87      4.10   53.00
##   10     De $610.001 a $670.000 mensuales liquidos   17    0.57      2.68   55.68
##   11     De $670.001 a $730.000 mensuales liquidos   19    0.64      3.00   58.68
##   12     De $730.001 a $800.000 mensuales liquidos   19    0.64      3.00   61.67
##   13     De $800.001 a $890.000 mensuales liquidos   12    0.40      1.89   63.56
##   14     De $890.001 a $980.000 mensuales liquidos   14    0.47      2.21   65.77
##   15   De $980.001 a $1.100.000 mensuales liquidos   21    0.70      3.31   69.09
##   16 De $1.100.001 a $1.260.000 mensuales liquidos   13    0.44      2.05   71.14
##   17 De $1.260.001 a $1.490.000 mensuales liquidos    5    0.17      0.79   71.92
##   18 De $1.490.001 a $1.850.000 mensuales liquidos   10    0.34      1.58   73.50
##   19 De $1.850.001 a $2.700.000 mensuales liquidos    8    0.27      1.26   74.76
##   20          Mas de $2.700.000 mensuales liquidos    7    0.23      1.10   75.87
##   88                                    No sabe   49    1.64      7.73   83.60
##   99                                No responde  104    3.49     16.40  100.00
##   NA                                         NA 2350   78.75        NA      NA
```

Recodificación: imputación del promedio del rango. Se genera una variable nueva (ingreso_rango_prom) para luego poder comparar con la original:

```
data_n$ingreso_rango_prom <- recode(data_n$ingreso_rango, "
1=110000;
2=250000;
3=305000;
4=325000;
5=400000;
6=445000;
7=490000;
8=535000;
9=585000;
10=640000;
11=700000;
```

```
12=765000;
13=845000;
14=935000;
15=995000;
16=1180000;
17=1375000;
18=1670000;
19=2275000;
20=3500000;
c(88,99)=NA
")
```

```
frq(data_n$ingreso_rango_prom)
```

```
##
## # A continuacion le presentamos un listado de rangos de ingreso. Podria usted indi (x) <numeric>
## # total N=2984  valid N=481  mean=590374.22  sd=527060.41
##
##      val                                        label  frq raw.prc valid.prc cum.prc
##        1          Menos de $220.000 mensuales liquidos    0    0.00      0.00    0.00
##        2     De $220.001 a $280.000 mensuales liquidos    0    0.00      0.00    0.00
##        3     De $280.001 a $330.000 mensuales liquidos    0    0.00      0.00    0.00
##        4     De $330.001 a $380.000 mensuales liquidos    0    0.00      0.00    0.00
##        5     De $380.001 a $420.000 mensuales liquidos    0    0.00      0.00    0.00
##        6     De $420.001 a $470.000 mensuales liquidos    0    0.00      0.00    0.00
##        7     De $470.001 a $510.000 mensuales liquidos    0    0.00      0.00    0.00
##        8     De $510.001 a $560.000 mensuales liquidos    0    0.00      0.00    0.00
##        9     De $560.001 a $610.000 mensuales liquidos    0    0.00      0.00    0.00
##       10     De $610.001 a $670.000 mensuales liquidos    0    0.00      0.00    0.00
##       11     De $670.001 a $730.000 mensuales liquidos    0    0.00      0.00    0.00
##       12     De $730.001 a $800.000 mensuales liquidos    0    0.00      0.00    0.00
##       13     De $800.001 a $890.000 mensuales liquidos    0    0.00      0.00    0.00
##       14     De $890.001 a $980.000 mensuales liquidos    0    0.00      0.00    0.00
##       15   De $980.001 a $1.100.000 mensuales liquidos    0    0.00      0.00    0.00
##       16 De $1.100.001 a $1.260.000 mensuales liquidos    0    0.00      0.00    0.00
##       17 De $1.260.001 a $1.490.000 mensuales liquidos    0    0.00      0.00    0.00
##       18 De $1.490.001 a $1.850.000 mensuales liquidos    0    0.00      0.00    0.00
##       19 De $1.850.001 a $2.700.000 mensuales liquidos    0    0.00      0.00    0.00
##       20          Mas de $2.700.000 mensuales liquidos    0    0.00      0.00    0.00
##       88                                       No sabe    0    0.00      0.00    0.00
##       99                                  No responde    0    0.00      0.00    0.00
##   110000                                       110000   51    1.71     10.60   10.60
##   250000                                       250000   44    1.47      9.15   19.75
```

```
##   305000                                  305000   39    1.31      8.11   27.86
##   325000                                  325000   39    1.31      8.11   35.97
##   400000                                   4e+05   37    1.24      7.69   43.66
##   445000                                  445000   37    1.24      7.69   51.35
##   490000                                  490000   41    1.37      8.52   59.88
##   535000                                  535000   22    0.74      4.57   64.45
##   585000                                  585000   26    0.87      5.41   69.85
##   640000                                  640000   17    0.57      3.53   73.39
##   700000                                   7e+05   19    0.64      3.95   77.34
##   765000                                  765000   19    0.64      3.95   81.29
##   845000                                  845000   12    0.40      2.49   83.78
##   935000                                  935000   14    0.47      2.91   86.69
##   995000                                  995000   21    0.70      4.37   91.06
##  1180000                                 1180000   13    0.44      2.70   93.76
##  1375000                                 1375000    5    0.17      1.04   94.80
##  1670000                                 1670000   10    0.34      2.08   96.88
##  2275000                                 2275000    8    0.27      1.66   98.54
##  3500000                                 3500000    7    0.23      1.46  100.00
##       NA                                      NA 2503   83.88        NA      NA
```

Ok, pero vemos que quedaron las etiquetas asociadas a los valores de la variable original; esto en general no es problema para los modelos pero si para descriptivos, así que mejor se borran con `drop_labels`, de `sjlabelled`

```
data_n$ingreso_rango_prom <- sjlabelled::drop_labels(data_n$ingreso_rango_prom)
frq(data_n$ingreso_rango_prom)
```

```
##
## # A continuacion le presentamos un listado de rangos de ingreso. Podria usted indi (x) <numeric>
## # total N=2984  valid N=481  mean=590374.22  sd=527060.41
##
##      val  frq raw.prc valid.prc cum.prc
##   110000   51    1.71     10.60   10.60
##   250000   44    1.47      9.15   19.75
##   305000   39    1.31      8.11   27.86
##   325000   39    1.31      8.11   35.97
##    4e+05   37    1.24      7.69   43.66
##   445000   37    1.24      7.69   51.35
##   490000   41    1.37      8.52   59.88
##   535000   22    0.74      4.57   64.45
##   585000   26    0.87      5.41   69.85
##   640000   17    0.57      3.53   73.39
##    7e+05   19    0.64      3.95   77.34
##   765000   19    0.64      3.95   81.29
```

```
##   845000   12    0.40      2.49   83.78
##   935000   14    0.47      2.91   86.69
##   995000   21    0.70      4.37   91.06
##  1180000   13    0.44      2.70   93.76
##  1375000    5    0.17      1.04   94.80
##  1670000   10    0.34      2.08   96.88
##  2275000    8    0.27      1.66   98.54
##  3500000    7    0.23      1.46  100.00
##     <NA> 2503   83.88        NA      NA
```

Ahora como tercer paso, combinamos ambas variables con rowSums (sumar las dos variables en una tercera). Al sumar ambas con `rowSums`, las que tienen doble missing quedan con valor 0, por lo tanto luego los 0 se recodifican a `NA`

```
frq(data_n$ingreso_rango_prom)
```

```
##
## # A continuacion le presentamos un listado de rangos de ingreso. Podria usted indi (x) <numeric>
## # total N=2984  valid N=481  mean=590374.22  sd=527060.41
##
##      val  frq raw.prc valid.prc cum.prc
##   110000   51    1.71     10.60   10.60
##   250000   44    1.47      9.15   19.75
##   305000   39    1.31      8.11   27.86
##   325000   39    1.31      8.11   35.97
##    4e+05   37    1.24      7.69   43.66
##   445000   37    1.24      7.69   51.35
##   490000   41    1.37      8.52   59.88
##   535000   22    0.74      4.57   64.45
##   585000   26    0.87      5.41   69.85
##   640000   17    0.57      3.53   73.39
##    7e+05   19    0.64      3.95   77.34
##   765000   19    0.64      3.95   81.29
##   845000   12    0.40      2.49   83.78
##   935000   14    0.47      2.91   86.69
##   995000   21    0.70      4.37   91.06
##  1180000   13    0.44      2.70   93.76
##  1375000    5    0.17      1.04   94.80
##  1670000   10    0.34      2.08   96.88
##  2275000    8    0.27      1.66   98.54
##  3500000    7    0.23      1.46  100.00
##     <NA> 2503   83.88        NA      NA
```

```
data_n <- data_n %>% mutate(ing_comb = rowSums(select(., "ingreso_monto",
    "ingreso_rango_prom"), na.rm = TRUE))
```

```
data_n$ing_comb = recode(data_n$ing_comb, "0=NA")

names(data_n)

## [1] "sstatus"           "ingreso_monto"     "ingreso_rango"     "educacion"         "sexo"

summary(data_n$ing_comb)

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.      NA's
## 2.000e+04 2.800e+05 4.450e+05 2.157e+06 7.000e+05 4.000e+09       161
```

Con esto disminuimos los missings de ingreso a solo 161. Ahora nos queda solamente dividir esta variable por el numero de integrantes del hogar:

```
frq(data_n$nhogar)

##
## # Numero de miembros del hogar (x) <numeric>
## # total N=2984  valid N=2984  mean=3.09  sd=1.57
##
##   val frq raw.prc valid.prc cum.prc
##     1 439   14.71     14.71   14.71
##     2 764   25.60     25.60   40.32
##     3 701   23.49     23.49   63.81
##     4 589   19.74     19.74   83.55
##     5 295    9.89      9.89   93.43
##     6 116    3.89      3.89   97.32
##     7  44    1.47      1.47   98.79
##     8  24    0.80      0.80   99.60
##     9   5    0.17      0.17   99.77
##    10   3    0.10      0.10   99.87
##    11   2    0.07      0.07   99.93
##    13   1    0.03      0.03   99.97
##    15   1    0.03      0.03  100.00
##  <NA>   0    0.00        NA      NA
```

No se detectan missins, así que se procede:

```
names(data_n)

## [1] "sstatus"           "ingreso_monto"     "ingreso_rango"     "educacion"         "sexo"

data_n <- data_n %>% mutate(ing_comb_eq = ing_comb/nhogar)
```

*Generación de ingreso por categorías (quintiles/deciles)*

Para quintiles vamos a crear una variable llamada quint, aplicando la función `ntile` a la variable de ingreso inc_comb_eq

```
data_n <- data_n %>% mutate(quint = ntile(ing_comb_eq, 5))
frq(data_n$quint)
```

```
##
## # x <integer>
## # total N=2984  valid N=2823  mean=3.00  sd=1.41
##
##   val frq raw.prc valid.prc cum.prc
##     1 565   18.93     20.01   20.01
##     2 565   18.93     20.01   40.03
##     3 564   18.90     19.98   60.01
##     4 565   18.93     20.01   80.02
##     5 564   18.90     19.98  100.00
##  <NA> 161    5.40        NA      NA
```

Y para deciles:

```
data_n <- data_n %>% mutate(decil = ntile(ing_comb_eq, 10))
frq(data_n$decil)
```

```
##
## # x <integer>
## # total N=2984  valid N=2823  mean=5.50  sd=2.87
##
##   val frq raw.prc valid.prc cum.prc
##     1 283    9.48     10.02   10.02
##     2 282    9.45      9.99   20.01
##     3 282    9.45      9.99   30.00
##     4 283    9.48     10.02   40.03
##     5 282    9.45      9.99   50.02
##     6 282    9.45      9.99   60.01
##     7 283    9.48     10.02   70.03
##     8 282    9.45      9.99   80.02
##     9 282    9.45      9.99   90.01
##    10 282    9.45      9.99  100.00
##  <NA> 161    5.40        NA      NA
```